REMARKS

Claims 16-24 and 26-46 are pending, of which claims 16, 33, 45 and 46 are independent. No claims have been amended. Applicant respectfully requests reconsideration and withdrawal of the outstanding rejections in view of the comments set forth below.

I. Summary of Rejections

The Examiner rejects claims 33-44 and 46 under 35 U.S.C. §102(e) in view of U.S. Patent No. 6,937,257 B1 (Dunlavey). The Examiner rejects claims 16-24, 26-32 and 45 under 35 U.S.C. §103 in view of Dunlavey and U.S. Patent No. 5,475,851 (Kodosky). Applicant respectfully traverses the rejections as discussed below.

II. Claim Rejections – 35 U.S.C. §102

The Examiner alleges that "Claims 33-44 and 46 are rejected under 35 U.S.C. 102(e) as being clearly anticipated by U.S. Patent No. 6,937,257 B1 (Dunlavey)." (See office action, page 2). Applicant respectfully traverses the rejection for at least the reasons discussed below.

A. Claims 33 and 46

Independent claim 33 of the present application reads as follows:

"A method of mapping graphical block diagram block parameters in a graphical block diagram modeling environment, comprising:

receiving a plurality of user-defined block parameters;

processing the plurality of user-defined block parameters to produce a plurality of run-time block parameters;

pooling together like non-interfaced run-time block parameters to reuse data for the like non-interfaced run-time block parameters."

Claim 46 reads as follows:

"A medium for use in a graphical modeling environment on an electronic device, the medium holding instructions executable using the electronic device for performing a method of mapping graphical block diagram block parameters, the method comprising:

receiving a plurality of user-defined block parameters;

processing the plurality of user-defined block parameters to produce a plurality of run-time block parameters; and

pooling together like non-interfaced run-time block parameters to reuse data for the like non-interfaced run-time block parameters."

Dunlavey does not disclose or suggest "processing the plurality of *user-defined block parameters* to produce a plurality of run-time block parameters," as required by claims 33 and 46. The Examiner relies on the following excerpt of Dunlavey as disclosing "processing the plurality of user-defined block parameters":

> "This graphical user interface enables the user to enter units-specifying data for the objects using unit expressions, such as "{kg/L}." While the objects are placed and connected by a user, the objects are converted into an internal format, such as a parse tree, representing statements for the computational model under construction, and the units-specifying data is translated into multidimensional unit type data." (See column 3, lines 1-8).

Reliance on this excerpt is, however, misplaced. Dunlavey does not process *user-defined block parameters* to produce a plurality of run-time block parameters, as required by claims 33 and 46. Dunlavey states: "the *units-specifying data* is translated into multidimensional unity type data." (See column 3, lines 6-8). Dunlavey further states: "In one embodiment, there are five basic dimensions of physical units: (1) volume, (2) weight, (3) time, (4) amount, and (5) age. A variable may have a unit type specification that comprises any subset of these dimensions, including all five, or none at all." (See column 16, lines 52-56). Units-specifying data in Dunlavey are not the variables themselves but rather are a specification of the units corresponding to the variables. In contrast, user-defined parameters in claims 33 and 46 are the parameter's values to be used by the block diagram. Thus, Applicant submits that units-specifying data in Dunlavey and *user-defined parameters* in the present application are different entities.

Dunlavey does not disclose "pooling together like non-interfaced run-time block parameters to reuse data for the like non-interfaced run-time block parameters," as recited in claims 33 and 46. The Examiner alleges that Dunlavey discloses "the grouping or pooling together of like non-interfaced run-time block parameters to create a run-time parameter expression for use during modeling, wherein Figure 4 lists the expression "SetDiscrete" which includes a group of run-time block parameters that are grouped and share a commonality of belonging to this group." (See office action, page 3). The Examiner also later alleges that "Dunlavey does disclose examples of when like parameters are grouped or pooled together (column 24, lines 35-45). Dunlavey also discloses multivariate distribution blocks, which hold like parameters. The variables are reused within various expressions, the distribution block being reused multiple times (column 3, lines 8-11)." (See office action, page 10). Applicant respectfully disagrees and contends that Dunlavey fails to disclose "pooling together like non-interfaced run-time block parameters to reuse data for the like non-interfaced run-time block parameters," as required by claims 33 and 46 for the reasons set forth below.

Dunlavey does not disclose "pooling together *like non-interfaced run-time block parameters*." In Dunlavey, the SetDiscrete primitive is used to set a group of categorical variables that are jointly distributed. (See drawing sheet 5). The commonality among the entities acted on by SetDiscrete is that they are categorical variables that are jointly distributed. The Dunlavey reference is silent about further implementation details of the categorical variables other that their depiction in drawing sheets, and does not disclose that the categorical variables are non-interfaced run-time parameters. Thus Applicant submits that the Dunlavey patent does not disclose "pooling of *like non-interfaced run-time parameters*," as recited by claims 33 and 46.

Dunlavey fails to disclose that the aforementioned pooling is conducted "to *reuse data* for the like non-interfaced run-time block parameters," as required by the aforementioned feature of claims 33 and 46. Dunlavey discusses the use of the aforementioned SetDiscrete primitive in Fig. 4. (See drawing sheet 5). Dunlavey

further states "FIG. 4 is a table showing primitives for an internal parse tree data structure for use in translating model blocks into equations according to one embodiment." (See column 3, lines 46-48). The Dunlavey reference is, however, silent about the specific purpose of using the aforementioned SetDiscrete primitive, referenced by the Examiner. Applicant therefore submits that Dunlavey fails to disclose "pooling together like non-interfaced run-time block parameters to *reuse data* for the like non-interfaced run-time block parameters," as required by claims 33 and 46.

Since Dunlavey does not disclose or suggest the features of claims 33 and 46, Applicant respectfully requests that the 35 U.S.C. 102(e) rejection of claims 33 and 46 be withdrawn. Reconsideration and allowance of claims 33 and 46 is respectfully requested in view of the above remarks.

B. Claims 34-44

Claims 34-44 are dependent upon base claim 33. As such, claims 34-44 include all the features of claim 33. For at least the reasons in the foregoing arguments on claim 33, Applicant submits that Dunlavey fails to disclose all of the features of claims 34-44. Since the reference does not disclose or suggest the features of claims 34-44, a 35 U.S.C. 102(e) rejection of claims 34-44 is improper and should be withdrawn. Applicant respectfully requests that the 35 U.S.C. 102(e) rejection of claims 34-44 be withdrawn.

III. Claim Rejections – 35 U.S.C. §103

The Examiner alleges that "Claims 16-24, 26-32 and 45 are unpatentable under 35 U.S.C. 103(a) in view of Dunlavey and U.S. Patent No. 5,475,851 (Kodosky et al.), herein referred to as "Kodosky '851." (See office action 3, page 5). Applicant respectfully traverses the rejection.

A. Claims 16 and 45

Independent claim 16 of the present application reads as follows:

"A method of mapping graphical block diagram block parameters in a graphical block diagram modeling environment, comprising:

receiving a user-defined block parameter representing a value to be used during block diagram execution; and

processing the user-defined block parameter to produce a run-time block parameter, wherein the run-time block parameter reduces memory requirements for executing a block diagram."

Claim 45 reads as follows:

"A medium for use in a graphical modeling environment on an electronic device, the medium holding instructions executable using the electronic device for performing a method of mapping graphical block diagram block parameters, the method comprising:

receiving a user-defined block parameter representing a value to be used during block diagram execution; and

processing the user-defined block parameter to produce a run-time block parameter for use during execution, wherein the run-time block parameter reduces memory requirements for executing a block diagram."

Dunlavey and Kodosky '851, alone or in any reasonable combination, fail to teach or suggest "processing the user-defined block parameter to produce a run-time block parameter," as required by claims 16 and 45. The Examiner alleges that "Dunlavey discloses processing the user-defined block parameter to produce a run-time block parameter for use during modeling," (see office action, page 6), and points to the language quoted above at column 3, lines 1-8 of Dunlavey. Applicant respectfully disagrees. Dunlavey does not teach or suggest "processing the user-defined block parameter to produce a run-time block parameter," as recited by claims 16 and 45. In contrast to the claimed invention, Dunlavey discusses conversion of objects into an internal format, such as a parse tree, and translation of units-specifying

data into multidimensional unit type data. Dunlavey fails to disclose processing of the *user-defined block parameter* to produce a run-time block parameter. The addition of Kodosky '851 fails to cure these deficiencies for the reasons set forth below.

Kodosky '851 states

> "The present invention comprises local and global variable capabilities which enable a user to more effectively create block diagram programs and model processes in a graphical data flow programming environment. The local and global variables include associated panel controls which allow a user to read or adjust the values of these variables during program execution.
>
> A global variable is created by placing a global variable icon in a block diagram and creating an associated global variable panel. The global variable panel is not associated with a VI. Therefore, a global variable is not created in a functional VI, but rather is created with its own panel not associated with a VI whose only purpose is to facilitate the implementation of a global variable. The user then defines the data type of the global variable and assigns a name to the global variable. Therefore, global variables are not required to be implemented as shift registers in a loop. This greatly reduces the storage and execution overhead associated with creating and using global variables." (See column 6, lines 13-32).

Kodosky '851 also fails to teach or suggest "processing the user-defined block parameter to produce a run-time block parameter," as required by claims 16 and 45. Applicant disagrees with the Examiner's allegation that "Kodosky discloses using global parameters in a graphical block diagram that is compiled and then executed, thereby the parameters representing run-time block parameters (column 1, lines 28-32)." (See office action, page 6). The language pointed to by the Examiner at column 1, lines 28-32 states "The present invention relates to graphical systems for creating and executing data flow programs, and more specifically to method and apparatus for improved local and global variable capabilities in a graphical data flow programming environment." The language cited by the Examiner summarizes the Kodosky '851 invention and discloses its global variable capabilities. The reference does not disclose processing block parameters (or for that matter global variables) to produce run-time parameters. Rather, claims 16 and 45 of the present application recite a run-time block parameter, not global variables, in implementing block diagram models. The Kodosky '851 reference therefore does not teach or suggest "processing the user-defined block parameter to produce a run-time block parameter." Based on the foregoing arguments, Applicant submits that Dunlavey and Kodosky '851, alone or in

any reasonable combination, fail to teach or suggest "processing the user-defined block parameter to produce a run-time block parameter," as required by claims 16 and 45.

Neither Dunlavey nor Kodosky '851, alone or in any reasonable combination, teach or suggest "processing the user-defined block parameter to produce a run-time block parameter, wherein the run-time block parameter reduces memory requirements for executing a block diagram," as required by claims 16 and 45. The Examiner alleges that "Kodosky further teaches that *the use of global variables* reduces memory requirements for execution of the block diagram, with the storage and execution process becoming more efficient." (See office action 3, page 6). Applicant respectfully disagrees with the Examiner for the reasons set forth below, and submits that Kodosky '851 fails to teach or suggest that "the *run-time block parameter* reduces memory requirements for executing a block diagram," as required by the aforementioned feature of claims 16 and 45.

Kodosky '851 discusses memory requirements as only pertaining to global variables and only in comparison to the U.S. Patent No. 4,901,221 (Kodosky et al.) prior art, herein referred to as Kodosky '221. The background section of Kodosky '851 states "The method described in Kodosky et al. provided a way of creating "global variables" which are implemented as shift registers.... This resulted in numerous inefficiencies, such as memory storage overhead and execution overhead. According to the present invention, a global variable corresponds to a control on a global variable panel, i.e., global variables are implemented simply as a control and are not associated with a shift register or loop function. Thus, implementing a global variable as a control considerably reduces both the storage and execution overhead." (See column 59, lines 1-15).

The reduction in memory requirements in Kodosky '851 is specifically attributed to the use of *controls* in implementing *global variables*. In contrast, claims 16 and 45 of the present application state "the *run-time block parameter* reduces memory requirements for executing a block diagram." Claims 16 and 45 thus attribute the reduction in memory requirements in the claimed invention to the run-

time block parameter, not to the implementation of any global variables as per Kodosky '851. Thus, Kodosky '851 fails to teach or suggest that "the *run-time block parameter* reduces memory requirements for executing a block diagram," as required by claims 16 and 45.

Moreover, there is no showing of a motivation to combine Dunlavey with Kodosky '851. Dunlavey and Kodosky '851 suggest two very different environments that do not appear to be compatible. The Examiner has provided no firm basis that one skilled in the art would be motivated to combine the teachings of the two references and be able to produce an operative combination. For example, Kodosky '851 concerns a data flow programming environment, whereas Dunlavey does not appear to support data flow programming at all. Dunlavey appears to be finely focused on drug modeling and employs specialty blocks that are particular to drug modeling. Kodosky '851 discusses modeling instruments as virtual instruments and does not concern itself with drug modeling.

In light of the foregoing reasons, Applicant respectfully submits that Dunlavey and Kodosky '851, alone or in any reasonable combination, fail to teach or suggest all of the features of claims 16 and 45. Since the references do not teach or suggest the features of claims 16 and 45, a 35 U.S.C. 103(a) rejection of claims 16 and 45 in view of Dunlavey and Kodosky '851 is improper and should be withdrawn. Applicant respectfully requests that the 35 U.S.C. 103(a) rejection of claims 16 and 45 be withdrawn. Reconsideration and allowance of claims 16 and 45 is respectfully requested in view of the above remarks.

B. Claims 17-24 and 26-32

Claims 17-24 and 26-32 are dependent upon claim 16. In view of the foregoing arguments about claims 16 and 45 in light of Dunlavey and Kodosky, Applicant submits that Dunlavey and Kodosky fail to teach or suggest all of the features of claims 17-24 and 26-32. Since the references do not teach or suggest the features of claims 17-24 and 26-32, a 35 U.S.C. 103(a) rejection of claim 17-24 and 26-32 is improper and should be withdrawn. Applicant respectfully requests that the

35 U.S.C. 103(a) rejection of claims 17-24 and 26-32 be withdrawn. Applicant requests the reconsideration and allowance of claims 17-24 and 26-32 in view of the above remarks.

CONCLUSION

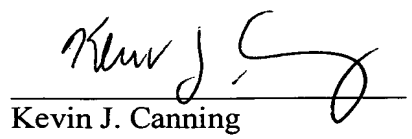
In light of the aforementioned arguments, Applicant submits that Dunlavey and Kodosky, alone or in combination, fail to disclose, teach or suggest the patentable features of the invention, and contends that the claimed invention is novel and non-obvious in view of Dunlavey and Kodosky.

Please charge any shortage or credit any overpayment of fees to our Deposit Account No. 12-0080, under Order No. MWS-072. In the event that a petition for an extension of time is required to be submitted herewith, and the requisite petition does not accompany this response, the undersigned hereby petitions under 37 C.F.R. §1.136(a) for an extension of time for as many months as are required to render this submission timely. Any fee due is authorized to be charged to the aforementioned Deposit Account.

In view of the above comments, Applicant believes that the pending application is in condition for allowance and urges the Examiner to pass the claims to allowance. Should the Examiner feel that a teleconference would expedite the prosecution of this application, the Examiner is urged to contact the Applicant's attorney at (617) 227-7400.

Date: November 8, 2006

Respectfully submitted,



Kevin J. Canning
Registration No.: 35,470
LAHIVE & COCKFIELD, LLP
One Post Office Square
Boston, Massachusetts 02109-2127
(617) 227-7400
(617) 742-4214 (Fax)
Attorney/Agent For Applicant